MACHINE CODE MADE EASIER

BOOKREV (LTF)

"A Practical Guide to Machine Language Programming on the Timex/Sinclair 1000 and 1500 (and ZX81)"

by David B. Wood
Siriusware
6 Turning Mill Road
Lexington, MA 02173
Price: $14.95

Reviewed by F. Nachbaur

If you don't want to run this, let me know & I'll send it to someone else.

Don't let the length of the title of this book put you off. This 250-page tutorial and reference work is by far the friendliest, most useful book of its kind that I've seen. Reading is easy, the author's style is conversational yet without the forced attempts at corny humor, and goofy cartoons, that tend to plague other tutorial books.

"MLP-TS" was slated for publication through an established publishing house, but was dropped when Timex went belly-up. Rather than let the book die, the author decided to publish it himself. I'm very glad that he did, as it is a very significant contribution to the T/S world.

The best way to describe the book is as a hybrid between a machine-code tutorial, an applications volume on the ZX ROM, and a machine-code reference work. It is specifically for the ZX81/TS1000 and TS1500. However, since much of the book is applicable to any Z80-based machine, 2068ers will also find much of interest here. The tutorial aspect extensively discusses the Z80 commands, and includes a liberal spattering of sample routines. The author places a great deal of emphasis on the efficient use of the ROM, so that your code need not "re-invent the ZX." Again, sample routines illustrate actual applications of many of the ROM calls. As a reference work, MLP-TS contains tables, a ROM-call library, and other information essential to the machine language programmer. Organization is impeccable throughout, aided by a comprehensive index.

This book, alone, will not turn you into an ace machine-coder. That can only happen through practise, experimentation, and more practise. The author makes it clear that his purpose is NOT to teach you how to write machine-code. However, it WILL get you started, and take you further than any other single volume on the subject. If you found Toni Baker's and Ian Logan's books confusing and a bit "heavy," MLP-TS will come as welcome relief. Though it does not replace the other works, it contains much of the same information in a much more palatable form. After getting through MLP-TS, you just might find the others more confrontable.

A very strong feature of the book is its extensive coverage of the floating-point calculator. Mr. Wood manages to thoroughly convince you how easy it really is to use this amazing part of the ROM. If you are presently limited to the rudimentary two-byte integer math supplied in the Z80, then this book will open up new vistas in your ML programming efforts. A table of all the FP calculator commands is included, along with a description of each of the FORTH-like commands. Yes, now you can easily do five-byte floating-point calculations within your machine code. Since the calculator is also used to handle string parameters, knowledge of its operation will also give you a firm handle on strings and string variables. With the exception of actual locations of the ROM calls, this aspect of the book is directly applicable to the Spectrum and 2068 as well.

The first chapter of the book is titled, "Bridging the Gap." This concept is the watch-word throughout the entire book. It appears that the author's prime purpose is to show you how to integrate the powerful Sinclair BASIC with the "native tongue" of the Z80, which is just as powerful in completely different ways. MLP-TS shows you how to get the best of both worlds; the precision and friendliness of BASIC with the speed and compactness of machine-code.

From there, the book discusses "Locating Machine-Code," summarizing the various ways in which your machine-code can be physically integrated with the other components of your programming environment. Chapter 3 is on "Software Tools;" to work with machine-code you will obviously need some sort of assembler and disassembler, and a way in which to move, relocate and otherwise manipulate your code segments.

To this end, a complete assembler, entirely in BASIC, is provided in one of the Appendices! Being an avowed Hot-Z user, I did not actually use this assembler, so I can't objectively evaluate this part of the book. However, on looking it over, it certain appears usable; if you find Hot Z too powerful (confusing) and ZXAS/ZXDB too simple (cumbersome), then the "Wood Assembler" might be just the ticket for you. (And the price is certainly right!)

The next chapter, "Organization of Memory," expands on and makes sense of the ZX/TS manual's chapter of the same name. "Finding Your Way Around BASIC" describes how BASIC lines are interpreted, and introduces the important concepts of the BASIC parser (CH_ADD pointer) and related topics. Starting with this chapter, short but useful (and highly educational) machine-code routines begin to make their appearance. Chapter 6 talks about "Getting At the Variables," but also introduces some extremely powerful ROM calls to manipulate the calculator stack. This paves the way for subsequent chapters on communication between BASIC and ML, use of the FP calculator, and related topics. Included are diagrams of the format of the different kinds of variables, to clarify the "bare-bones" treatment of the subject in the ZX/TS manual.

A short but informative chapter on "FAST and SLOW Mode" helps to clarify this rather unique aspect of the ZX/TS. This is followed by Chapter 8, "Manipulating Memory Space." This provides powerful techniques for making room anywhere in memory, reclaiming memory, and changing RAMTOP. Defining and erasing variables is also covered; the latter topic can be especially useful, since no provision for doing so in included in the machine's BASIC.

A short chapter on "Aborting" discusses the use of the BREAK key and the error restart to avoid getting stuck in your machine-code routines during writing/debugging. This is followed by a comprehensive chapter on "Writing to the TV Screen," which tells you everything-you-ever-wanted-to-know about the display file. Included are ways of printing integers and floating-point numbers, using the print restart, and direct display-file control using block moves and swaps. PLOT and UNPLOT, which have traditionally been difficult to handle from machine-code, are shown to be almost absurdly simple; a single flag byte is used to control PLOT vs. UNPLOT! Previous articles completely missed this point!

Chapter 11, "Communication and Passing Parameters" begins what is (for me) the meatiest part of the book. Using concepts presented in this section enabled me to develop an extremely flexible parameter-passing system for my "JOBASIC" extension for the Oliger VDP board. If your ML programming efforts to date resulted in long lists of USRs and PEEKs and POKEs, this chapter can help to simplify matters immensely. Chapter 12 is a very comprehensive treatment of the ZX/TS system for keyboard scanning. Included is a machine-code variant of INKEY$, which is extremely fast and will prove highly useful for text entry from machine-code. The KBD and DATA lines will now actually make sense to you!

The next two chapters, "Scanning BASIC" and "The Floating Point Calculator" further develop concepts introduced earlier. The material here is nothing short of phenomenal. Chapter 15, "The INs and OUTs of Cassette READ/WRITE" is another remarkable piece of work; using the information here, you could write your own customized tape routines to fit any need. Included is a complete routine to read ANY tape (or tape portion) recorded in Sinclair format.

The last chapter, "Programming and Debugging Hints" is a lot more than the title implies. It is here that the actual use of Z80 code is addressed in depth. Though it's the last "chapter," it's in reality a lead-in to the second portion of the book - the Appendices. These appendices take up over 1/3 of the bulk of the text, and almost deserve to be book in their own right. Appendix I is on the Z80 Microprocessor. In 20 pages, this manages to convey as much useful information on the Z80 as some stand-alone books I've seen!

Appendix II, "Z80 Operations," summarizes all the Z80 commands in tabular form. Appendix III, "Z80 Flags," helps to make sense of the sophisticated flag system built into the Z80. Again, though it is only two pages long, it summarizes what effect the various commands have on the flags far more understandably that entire chapters of books like the Rodney Zaks "Bible."

Appendix IV is the assembler mentioned previously. Appendix V is a table of the System Variables, along with their function and utility. Appendix VI, "A Catalog of ROM Routines," is a real gem. 98 different routines are described, including their entry points (in hex and decimal), utility, what happens to the registers and flags, and any additional information needed to make effective use of them. They are neatly organized into categories, to make it easy to find the suitable routine for a particular task.

Appendix VII is a handy hex/decimal conversion chart. The book ends with Appendix VIII, a bibliography for further study, followed by the index.

The book has very few shortcomings. None of them are even remotely "fatal." The section on the Z80 commands only briefly touches on the use of the shift and rotate commands, though a comprehensive table of diagrams is included. The BCD commands (RLD, RRD and DAA) are not discussed; perhaps this is because they are unlikely to be useful unless you intend to "re-invent the ZX." The Z80 interrupts are shrugged off with the admonition, "Don't mess with the interrupts." (For the most part, this is sound advice; however, if your machine has alternate arrangements for the video display, the interrupts can be a powerful tool. A discussion of these would have made this section of the book complete.) Also, the HALT command is treated as a "no-no," even though it can be useful in SLOW mode to insure that the subsequent code is only executed after a TV scan has been completed.

A few (very few) ROM routines that I have found useful were not included in the "Catalog." The text was printed on a NLQ dot-matrix, and there is no trouble with readability; however, a better look would have resulted from a daisy-wheel, with right-justified text. The routine listings and most of the tables were produced on a "standard" dot-matrix, and a few of them came out a little "thin." Some aspects of the FP calculator (such as Chebyshev polynomials) are only briefly touched upon. Perhaps this is just as well; not everyone will be interested in this. Those that are can refer to the works referenced in the text and the bibliography.

MLP-TS is the closest thing to a perfect "10" that I've seen. Any of its many component parts are, alone, worth the purchase price. My recommendation should be obvious by now; BUY IT!